

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

132 AF
IFN \$

Application of: **Bahrs et al.**

Serial No.: **09/430,861**

Filed: **October 29, 1999**

For: **Method and Apparatus in a Data Processing System for the Separation of Role-Based Permissions Specification from its Corresponding Implementation of its Semantic Behavior**

§
§
§
§
§
§

Group Art Unit: **2132**

Examiner: **Gurshman, Grigory**

Attorney Docket No.: **AUS990339US8**

Certificate of Mailing Under 37 C.F.R. § 1.8(a)

I hereby certify this correspondence is being deposited with the United States Postal Service as First Class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on August 16, 2004.

By:

Amelia C. Turner
Amelia C. Turner

35525

PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

TRANSMITTAL DOCUMENT

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:
ENCLOSED HERewith:

- Appellant's Brief (in triplicate) (37 C.F.R. 1.192); and
- Our return postcard.

A fee of \$330.00 is required for filing an Appellant's Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

Respectfully submitted,

Duke W. Yee

Duke W. Yee

Registration No. 34,285

YEE & ASSOCIATES, P.C.

P.O. Box 802333

Dallas, Texas 75380

(972) 367-2001

ATTORNEY FOR APPLICANTS



Doc No. AUS990339US8

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Bahrs et al.**

Serial No.: **09/430,861**

Filed: **October 29, 1999**

For: **Method and Apparatus in a Data
Processing System for the Separation
of Role-Based Permissions
Specification from its Corresponding
Implementation of its Semantic
Behavior**

§
§
§
§
§
§
§

Group Art Unit: **2132**

Examiner: **Gurshman, Grigory**

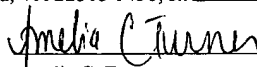
**Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450**

**ATTENTION: Board of Patent Appeals
and Interferences**

Certificate of Mailing Under 37 C.F.R. § 1.8(a)

I hereby certify this correspondence is being deposited with the
United States Postal Service as First Class mail in an envelope
addressed to: Commissioner for Patents, P.O. Box 1450,
Alexandria, VA 22313-1450, on **08.16.04**.

By:


Amelia C. Turner

08/20/2004 AADDF01 00000004 090447 09430861
01 FC:1402 330.00 DA

APPELLANT'S BRIEF (37 C.F.R. 1.192)

This brief is in furtherance of the Notice of Appeal, filed in this case on June 14, 2004.

The fees required under § 1.17(c), and any required petition for extension of time for filing this
brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

This brief is transmitted in triplicate. (37 C.F.R. 1.192(a))

REAL PARTIES IN INTEREST

As reflected in the Assignment recorded on October 29, 1999, at Reel 010358, Frame 0189, the present application is assigned to International Business Machines Corporation, the real party in interest.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

Claims 187-226 stand finally rejected as noted in the Final Office Action mailed April 21, 2004.

STATUS OF AMENDMENTS

Applicants' Response to Office Action, transmitted on February 23, 2004, has been entered. Applicants have not filed a response to the Final Office Action.

SUMMARY OF INVENTION

Applicants' claims describe a method for managing permissions in an application. User input is received changing a security level for the application. The user input is received at a container that is handled by a view controller. An application mediator that is included in a plurality of application mediators creates one or more view controllers. These view controllers include the

view controller that handles the container at which the user input was received. These one or more view controllers are created to complete a function of the application mediator. A view event is generated by the view controller. The view event describes the user input that was received at the container. The view event is received at the application mediator. Responsive to the application mediator receiving the view event, a request event is sent from the application mediator to the view controller. The application mediator receives a permission corresponding to the security level that was changed according to the user input. The permission alters an item in the application.

According to other independent claims, Applicants' claims describe receiving a user input at a container that is handled by a view controller. The user input requests a change in permission in the application. An application mediator creates the view controller to complete a function of the application mediator. A view event that describes the user input is generated by the view controller. The application mediator receives the view event. The application mediator then retrieves a set of permissions corresponding to the user input. The set of permissions is sent to the view controller from the application mediator. The content is selectively altered within the container using the set of permissions.

According to other independent claims, Applicants' claims describe a data processing system that comprises a view controller that handles presenting a container, handles input to the container, generates a view event in response to user input, and alters content in the container in response to a change in permissions. The view controller is created by an application mediator. The application mediator creates this and other view controllers to complete a function of the application mediator. The application mediator receives the view event, obtains permissions for content in response to the view event indicating a request to change permissions, and sends the permissions to the view controller to set the permissions in the view controller.

ISSUES

Are the Examiner's rejections of claims 187-192, 194-198, 200-204, 206-213, 215-219, and 221-226 under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent 6,108,583 issued to *Schneck* in view of U.S. Patent 4,813,653 issued to *Anderl*, and claims 193, 199, 205, 214, and 220 as being unpatentable over *Schneck* in view of *Anderl* and further in view of U.S. Patent 6,185,684 issued to *Pravetz* well founded?

GROUPING OF CLAIMS

For the purposes of this appeal, claims 187-226 stand or fall together as one group.

ARGUMENT

The Examiner rejected claims 187-192, 194-198, 200-204, 206-213, 215-219, and 221-226 under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent 6,108,583 issued to *Schneck* in view of U.S. Patent 4,813,653 issued to *Anderl*. This position is not well founded.

Applicants claim a view controller that is created by an application mediator. The application mediator creates one or more view controllers including this view controller. The view controller handles a container. User input that changes a security level is received at the container that is handled by the view controller. The view controller generates a view event that describes the user input. The view event is received by the application mediator that created the view controller. The application mediator then sends a request event back to the view controller that generated the view event. The application mediator receives a permission corresponding to the security level where the permission alters an item in the application.

Thus, Applicants claim user input being received at a container that is handled by a view controller. The view controller generates a view event and sends the view event to the application mediator that created the view controller. The application mediator creates a request event in response to the application mediator receiving the view event. This request event is then sent back to the view controller that generated the view event.

The Examiner rejects these claims as being unpatentable over *Schneck* in view of *Anderl*. The Examiner states that *Schneck* teaches a view controller by describing an “Authentication Header Generator” (reference 123 in Figure 1), and teaches an application mediator by describing an “Authentication Header Decomposer” (reference 146 in Figure 1).

Schneck teaches an Authentication Header Generator 123 that is located in one data processing system that is the transmitting host system. *Schneck* also teaches an Authentication Header Decomposer 146 that is located in a separate data processing system that is the receiving host system. A plurality of data packets are generated by a sending host computer system that includes an authentication data block with an authentication header that contains the actual

security configuration and a signature.

A receiving host computer system receives the data packets that include the authentication data block having an authentication header and a signature. The authentication header is then decomposed such that the fields are separated from the data packet.

Therefore, *Schneck* teaches two computer systems. One system includes a Generator 123 for generating authentication headers for data packets. These data packets along with the authentication headers are transmitted to a second computer system. This second, separate computer system then uses its Decomposer 146 to decompose the authentication headers.

The Examiner appears to believe that *Schneck* teaches the view controller by teaching Authentication Header Generator 123 and the application mediator by teaching the Authentication Header Decomposer 146. These devices taught by *Schneck* do not operate in a manner as claimed by Applicants.

The Examiner states *Schneck* does not teach the Decomposer 146 creating a plurality of Generators 123. The Examiner goes on to say that *Schneck* is used for teaching that the Decomposer 146 produces a plurality of authentication headers. Examiner's Final Action, page 2, paragraph 5.

The Generator 123 generates authentication headers for packets that are sent from the host system that includes the Generator 123. These packets are then sent to the receiver which includes the Decomposer 146. If, as the Examiner states, the Generator 123 is the view controller, then the Decomposer 146 must create the Generator 123. In addition, since Applicants claim the application mediator creates one or more view controllers, *Schneck* must teach that the Decomposer creates one or more Generators 123. *Schneck* does not teach that the Decomposer 146 creates the Generator 123, and certainly does not teach that the Decomposer 146 creates one or more Generators 123.

Applicants claim the view controller is created by the application mediator. *Schneck* does not teach the Decomposer 146 creating the Generator 123. In fact, these devices are located in two different machines. Nothing in *Schneck* describes, teaches, or suggests the Decomposer 146 creating the Generator 123.

Applicants claim the application mediator creates one or more view controllers including the view controller that handles the container at which the user input is received. Nothing in *Schneck* teaches this. The Decomposer 146 of *Schneck* does not create one or more Generators

123.

Applicants claim the view controller generating a view event that is received by the application mediator that created the view controller where the view event describes the user input received at the container handled by the view controller. The Generator 123 generates authentication headers and sends them to the Decomposer 146. However, as described above, the Generator 123 is not created by the Decomposer 146. Further, nothing in *Schneck* teaches the Generator 123 handling a container at which the user input was received. Thus, *Schneck* does not teach the view controller generating a view event that is received by the application mediator that created the view controller where the view event describes the user input received at the container handled by the view controller.

Applicants claim the view controller handling a container at which the user input is received. The Examiner states that this is taught by *Schneck* by the user input 129 being received by the Generator 123. The Examiner does not state what part of *Schneck* teaches a container that is handled by a view controller. Since the Examiner believes the Generator 123 teaches a view controller, the Generator 123 must handle a container at which the user input is received. Nothing in *Schneck* teaches the Generator 123 handling any container at which the user input is received.

The Examiner states that the limitation “generating a view event, by the view controller, describing the user input” is met by the information shown on the display device (136 in Fig. 1). Examiner’s Final Action page 3, paragraph 6. Thus, the Examiner appears to be arguing that the view event of Applicants’ claims is met by the display device 136 of *Schneck*.

Applicants claim the view controller generating a view event that describes the user input. This view event is received by the application mediator. Since the Examiner argues that the Decomposer 146 is the application mediator, a view event must be received at the Decomposer 146 that describes the user input. The Examiner does not state that the Decomposer 146 receives such a view event. In fact, the Examiner states that a view event is received by display device 136 which is not what the Examiner relies on as teaching the application mediator.

Applicants claim the application receiving the view event and then sending a request event back to the view controller that generated the view event. *Schneck* does not teach the Decomposer 146 receiving an authentication header and then sending a request event to the Generator 123. The Examiner states that *Schneck* teaches this feature by teaching a request from

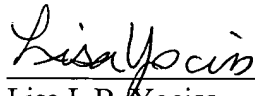
unit 146 to a security monitor 169. This is not what is claimed by Applicants. If the Decomposer 146 is indeed an application mediator as the Examiner believes, the Decomposer 146 would have to send a request event back to Generator 123 and not to security monitor 169. Nothing in *Schneck* teaches an application mediator receiving a view event and then sending a request event to the view controller that sent the view event. The Examiner does not point to any part of *Schneck* that teaches the Decomposer 146 sending a request event back to Generator 123. The Examiner merely states that Decomposer 146 generates an event that is sent to a completely different device, i.e. security monitor 169.

Claim 207 describes “a plurality of view controllers, wherein each view controller within the plurality of view controllers handles a container and wherein the container includes content, which is alterable by permissions, and wherein the application mediator handles permissions for the plurality of view controllers”. The Examiner rejects this claim as being unpatentable over *Schneck* in view of *Anderl*. However, the Examiner does not point to any part of either reference which discloses a plurality of view controllers.

The Examiner rejected claims 193, 199, 205, 214, and 220 as being unpatentable over *Schneck* in view of *Anderl* and further in view of U.S. Patent 6,185,684 issued to *Pravetz*. This position is not well founded.

The Examiner states that the combination of *Schneck* and *Anderl* does not teach the permission being a set of key/value pairs, and uses *Pravetz* to supply this missing feature. The combination of *Schneck*, *Anderl*, and *Pravetz* does not describe, teach, or suggest Applicants’ claims. The combination does not describe, teach, or suggest a view event, view controller, application mediator, or the interaction of these elements as claimed by Applicants.

Applicants' claims are patentable over the cited prior art. The combination of references does not describe teach or suggest a view controller that handles a container at which user input is received, an application that creates the view controller, the view controller generating a view event that is received by the application mediator, or the application sending a request event back to the view controller that generated the view event.



Lisa L.B. Yociss
Reg. No. 36,975
Yee & Associates, P.C.
PO Box 802333
Dallas, TX 75380
(972) 367-2001

APPENDIX OF CLAIMS

The text of the claims involved in the appeal reads:

187. A method in a data processing system for managing permissions in an application, the method comprising the data processing system implemented steps of:

receiving a user input changing a security level for the application at a container that is handled by a view controller;

creating, by an application mediator that is included in a plurality of application mediators that exist in said system, one or more view controllers including said view controller to complete a function of said application mediator;

generating a view event, by said view controller, describing the user input;

receiving the view event at said application mediator;

responsive to receiving the view event by said application mediator, sending a request event from said application mediator to said view controller that was created by said application mediator; and

receiving at the application mediator, a permission corresponding to the security level, wherein the permission alters an item in the application.

188. The method of claim 187, wherein the item is content in the container and further comprising:

sending the permission to the view controller, wherein the permission selectively alters content within the container.

189. The method of claim 187, wherein the item is a function in the application mediator.

190. The method of claim 187, wherein the user input changing the security level is a user login to the application.

191. The method of claim 187, wherein the content altered using the set of permissions is an enablement of a function.

192. The method of claim 187, wherein the content altered using the set of permissions is a disablement of a function.

193. The method of claim 187, wherein the set of permissions is a set of key/value pairs, wherein a key is used to identify content and a value is used to identify a setting for the content.

194. A method in a data processing system for managing permissions in an application, the method comprising the data processing system implemented steps of:

receiving a user input at a container that is handled by a view controller, wherein the user input requests a change in permissions in the application;

creating, by an application mediator that is included in a plurality of application mediators that exist in said system, one or more view controllers including said view controller to complete a function of said application mediator;

generating a view event, by said view controller, describing the user input;

receiving the view event at said application mediator;

responsive to receiving the view event by said application mediator, retrieving by the application mediator, a set of permissions corresponding to the user input;

sending the set of permissions to the view controller from said application mediator to said view controller that was created by said application mediator; and

selectively altering content within the container using the set of permissions.

195. The method of claim 194, wherein the step of selectively altering the content includes: enabling content within the container.

196. The method of claim 195, wherein the content enabled is a button.

197. The method of claim 194, wherein the step of selectively altering the contents includes: disabling content within the container.

198. The method of claim 196, wherein the content disabled is a text field.

199. The method of claim 194, wherein the set of permissions includes a plurality of key and value pairs, wherein a key in a key and value pair identifies an alterable item and a value in the key and value pair identifies a setting for the alterable item.

200. The method of claim 194, wherein the user input is a selection of a button.

201. The method of claim 194, wherein the set of permissions are associated with a user profile.

202. The method of claim 194, wherein the user input is a change in security.

203. A data processing system comprising:

a view controller, wherein the view controller handles presenting a container, handles input to the container, generates a view event in response to a selected user input, and alters content in the container in response to a change in permissions;

said view controller being created by an application mediator that creates one or more view controllers that include said view controller to complete a function of said application mediator; and

wherein the application mediator receives the view event, obtains permissions for content in the container in response to the view event indicating a request to change permissions for the content, and sends the permissions to the view controller that was created by said application mediator to set the permissions in the view controller.

204. The data processing system of claim 203, wherein the container is a panel.

205. The data processing system of claim 203, wherein the permissions are a set of key/value pairs, wherein a key in a key/value pair identifies alterable content and a value in the key/value pair identifies a setting for the alterable content.

206. The data processing system of claim 203, wherein permissions are associated with a user profile.

207. The data processing system of claim 203 further comprising:

a plurality of view controllers, wherein each view controller within the plurality of view controllers handles a container and wherein the container includes content, which is alterable by permissions, and wherein the application mediator handles permissions for the plurality of view controllers.

208. A data processing system for managing permissions in an application, the data processing system comprising:

first receiving means for receiving a user input changing a security level for the application at a container handled by a view controller;

an application mediator that is included in a plurality of application mediators for creating one or more view controllers including said view controller to complete a function of said application mediator;

generating means for generating a view event, by said view controller, describing the user input;

second receiving means for receiving the view event at said application mediator;

sending means, responsive to receiving the view event by said application mediator, for sending a request event from said application mediator to said view controller that was created by said application mediator; and

third receiving means for receiving at the application mediator, a permission corresponding to the security level, wherein the permission alters an item in the application.

209. The data processing system of claim 208, wherein the item is content in the container and further comprising:

sending means for sending the permission to the view controller, wherein the permission selectively alters content within the container.

210. The data processing system of claim 208, wherein the item is a function in the application mediator.

211. The data processing system of claim 208, wherein the user input changing the security level is a user login to the application.

212. The data processing system of claim 208, wherein the content altered using the set of permissions is an enablement of a function.

213. The data processing system of claim 208, wherein the content altered using the set of permissions is a disablement of a function.

214. The data processing system of claim 208, wherein the set of permissions is a set of key/value pairs, wherein a key is used to identify content and a value is used to identify a setting for the content.

215. A data processing system for managing permissions in an application, the data processing system comprising:

first receiving means for receiving a user input at a container handled by a view controller, wherein the user input requests a change in permissions in the application;

an application mediator that is included in a plurality of application mediators for creating one or more view controllers including said view controller to complete a function of said application mediator;

generating means for generating a view event, by said view controller, describing the user input;

second receiving means for receiving the view event at said application mediator;

retrieving means, responsive to receiving the view event, for retrieving by the application mediator, a set of permissions corresponding to the user input;

sending means for sending the set of permissions to the view controller that was created by said application mediator; and

altering means for selectively altering content within the container using the set of permissions.

216. The data processing system of claim 215, wherein the means of selectively altering the content includes:

enabling means for enabling content within the container.

217. The data processing system of claim 216, wherein the content enabled is a button.

218. The data processing system of claim 215, wherein the means of selectively altering the contents includes:

disabling means for disabling content within the container.

219. The data processing system of claim 217, wherein the content disabled is a text field.

220. The data processing system of claim 215, wherein the set of permissions includes a plurality of key and value pairs, wherein a key in a key and value pair identifies an alterable item and a value in the key and value pair identifies a setting for the alterable item.

221. The data processing system of claim 215, wherein the user input is a selection of a button.

222. The data processing system of claim 215, wherein the set of permissions are associated with a user profile.

223. The data processing system of claim 215, wherein the user input is a change in security.

224. A computer program product in a computer readable medium for managing permissions in an application, the computer program product comprising:

instructions for receiving a user input changing a security level for the application at a container handled by a view controller;

instructions for creating, by an application mediator that is included in a plurality of application mediators, one or more view controllers including said view controller to complete a function of said application mediator;

instructions for generating a view event, by said view controller, describing the user input;

instructions for receiving the view event at said application mediator;

instructions, responsive to receiving the view event, for sending a request event in response to the receiving the view event; and

instructions for receiving at the application mediator, a permission corresponding to the security level, wherein the permission alters an item in the application.

225. A computer program product in a computer readable medium for managing permissions in an application, the computer program product comprising:

instructions for receiving a user input at a container handled by a view controller, wherein the user input requests a change in permissions in the application;

instructions for creating, by an application mediator that is included in a plurality of application mediators, one or more view controllers including said view controller to complete a function of said application mediator;

instructions for generating a view event, by said view controller, describing the user input;

instructions for receiving the view event at said application mediator;

instructions, responsive to receiving the view event, for retrieving by the application mediator, a set of permissions corresponding to the user input;

instructions for sending the set of permissions to the view controller; and
instructions for selectively altering content within the container using the set of permissions.

226. A computer program product in a computer readable medium comprising:

first instructions for a view controller, wherein the view controller handles presenting a container, handles input to the container, generates a view event in response to a selected user input, and alters content in the container in response to a change in permissions;

said view controller being created by an application mediator that creates one or more view controllers that include said view controller to complete a function of said application mediator; and

second instructions for said application mediator, wherein the application mediator receives the view event, obtains permissions for content in the container in response to the view event indicating a request to change permissions for the content, and sends the permissions to the view controller to set the permissions in the view controller.